

Chapter 6

Configuring IPTV Services in an SDX Network

This chapter describes how the SDX network handles IP television (IPTV) services, and how to configure policies, services, and subscribers that support IPTV applications. It contains the following sections:

- Overview of IPTV Service Applications on page 117
- Installing the Sample IPTV Application on page 118
- Configuring the Sample IPTV Application on page 118
- Running the Sample IPTV Application on page 126

Overview of IPTV Service Applications

IPTV and multimedia service sessions are typically established in multiple phases that require changes to installed policies and authorized bandwidth while the service session remains active. To support IPTV sessions, the SAE allows changes to active service sessions. These changes include:

- Controlled bandwidth. If bandwidth demand increases, the authorization plug-in must authorize the change.
- Policy parameters. Only parameter substitution values can be changed. Policy parameters can include classifiers, such as destination address, and actions, such as rate-limit profiles.
- Session and idle timeouts. All attributes that can be set for initial service activation can be set for service session modifications.

You can integrate IP servers into an SDX-managed network so that the SDX software can receive network resource requests from IPTV applications to perform admission control and install policies. The SDX software can also set up and manage the initial label-switched paths (LSPs) between the video servers and the edge routers.

When the SAE activates a service session, it authorizes the session with authorization plug-ins; it may use the Admission Control Plug-in (ACP) application to perform call admission control (CAC) and allocate bandwidth; and it installs the policy required for the service on a router interface.

To support IPTV services, we provide a sample IPTV application that uses the extended capabilities of ACP and the SAE. You can use ACP to initialize and execute applications that are attached to congestion points. You can use the SAE router driver to set up and manage LSPs.

Installing the Sample IPTV Application

You must manually install the UMCiptv package on the server host to deploy the sample IPTV application.

```
pkgadd -d /cdrom/cdrom0/solaris UMCiptv
```



NOTE: The sample IPTV application is provided on the SDX application library CD.

For information about installing the sample IPTV application, see *SDX Application Library Guide, Chapter 1, Installing the SDX Applications*.

Configuring the Sample IPTV Application

The SDX application library provides a sample IPTV application with sample data that you must modify for your environment. The sample data provides the minimum requirements for demonstrating an IPTV application in the SDX-managed network. Figure 19 on page 120 illustrates a sample network configuration that contains Juniper Networks routers.

The sample network configuration has the following setup:

- The IPTV server connects to the M-series routing platform.
- The SAE manages the M-series routing platform.
- The SAE manages the E-series router and all its subscribers.
- The NIC is configured for the SAE.
- ACP is configured in backbone mode for the SAE managing the M-series routing platform.
- The M-series routing platform and the E-series router have MPLS configured for creating the LSP tunnel between them.

This sample configuration allows the sample IPTV application to:

- Start the IPTV service.

When the subscriber requests a video stream from the IPTV server, the IPTV service is activated by the SAE, and ACP activates a script service to create the LSP tunnel between the two routers over which the video stream is sent. ACP also updates the bandwidth usage for the LSP tunnel so the IPTV server can send the video stream to the subscriber.

- Stop the IPTV service.

When the subscriber is finished viewing the video stream from the IPTV server, the IPTV service is deactivated by the SAE, and ACP activates a script service to remove the tunnel.

- Manage tunnel bandwidth.

When tunnel bandwidth usage reaches the incremental threshold (80%) of the initial bandwidth value, ACP modifies the script service on the SAE to increase the bandwidth to the incremental bandwidth value. When the tunnel bandwidth usage reaches the decremental threshold (20%), ACP modifies the script service to decrease the bandwidth to the initial bandwidth value.

- Monitor tunnel state.

When the LSP tunnel is down, ACP can report the affected IPTV service sessions to the IPTV server.

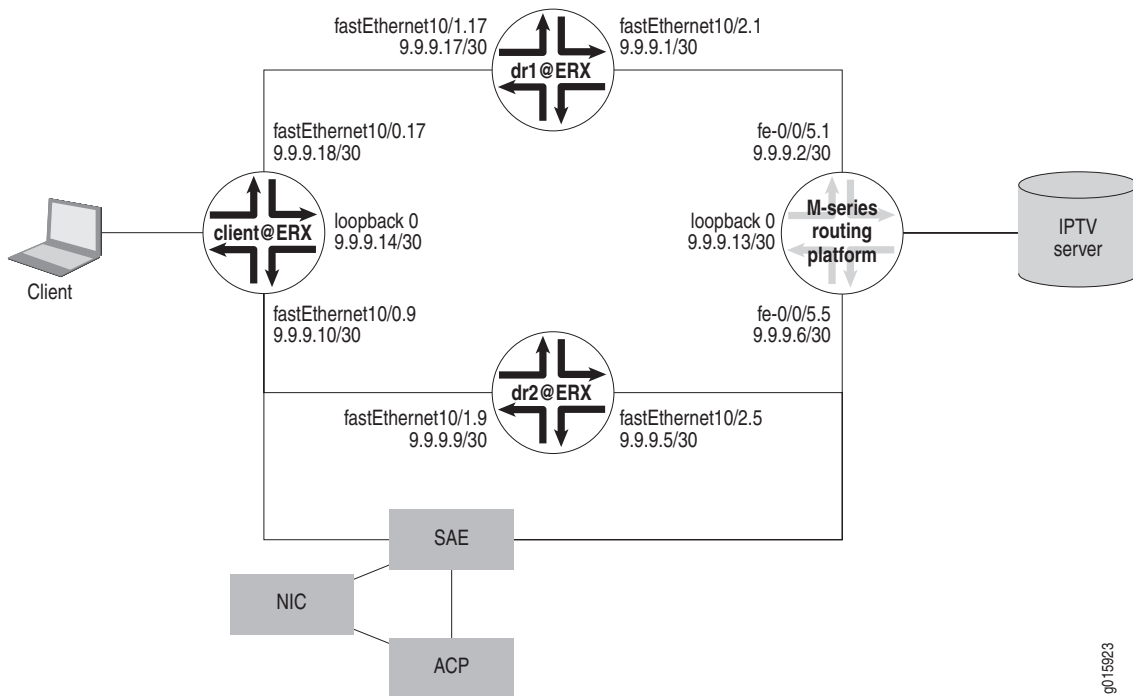
This section describes the configuration tasks for configuring the sample IPTV application.

- Setting Up the IPTV Network on page 120
- Configuring the SAE for the IPTV Application on page 121
- Configuring ACP for the IPTV Application on page 123
- Configuring the NIC for the IPTV Application on page 126

Setting Up the IPTV Network

Set up the network shown in Figure 19. The sample routing configuration is based on this network configuration.

Figure 19: Sample IPTV Network Configuration



To set up routing configuration from the IPTV client to the IPTV server, modify the following sample router scripts with interface names that match your network device names.

- */iptv/conf/network/client.scr*—Sets up the LSP tunnel endpoint (JUNOSe router)
- */iptv/conf/network/dr1.scr* — Sets up the path for the LSP tunnel (JUNOSe router)
- */iptv/conf/network/dr2.scr* — Sets up the path for the LSP tunnel (JUNOSe router)
- */iptv/conf/network/lspstest.scr* — Sets up the LSP tunnel starting point (JUNOS routing platform)

These scripts do not create the LSP tunnel. The tunnel service creates the LSP tunnel when it is activated. See *Configuring IPTV Subscribers and Services* on page 121.

Configuring the SAE for the IPTV Application

You must configure the SAE to:

1. Manage the routers.
2. Configure subscribers and services.
3. Configure ACP as an external plug-in for the SAE.
4. Configure event publishers.
5. Configure the NIC as an external plug-in.

Managing the Routers in an IPTV Network

You must configure the SAE to manage the JUNOSe router connected to the IPTV subscriber (client@ERX) and the JUNOS routing platform connected to the IPTV server. The JUNOSe router must be configured to manage the LSP tunnel so that the SAE can report information about the LSP tunnel to ACP.

To configure the SAE to manage the JUNOSe router or the JUNOS routing platform, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 3, Using JUNOSe Routers in the SDX Network* or *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 4, Using JUNOS Routing Platforms in the SDX Network*.

Configuring IPTV Subscribers and Services

Define the subscribers by importing the `/iptv/conf/ldap/subscribers.ldif` file from the installation directory (`/opt/UMC` by default) using an LDAP browser. You can then modify the following two subscribers with SDX Admin.

- `uniqueID = iptvSubscriber, ou = local, retailermame = IPTV, o = Users, o = umc`

The `iptvSubscriber` subscriber is the IPTV client.

- `uniqueID = iptvServer, ou = local, retailermame = IPTV, o = Users, o = umc`

The `iptvServer` subscriber is the IPTV server.

For detailed information about configuring subscribers, see *SDX Subscribers and Subscriptions Guide, Chapter 8, Configuring Subscribers and Subscriptions*.

Define the IPTV services by importing the `/iptv/conf/ldap/services.ldif` file from the installation directory (`/opt/UMC` by default) using an LDAP browser. You can then modify the services with SDX Admin.

- `serviceName = iptv, o = Services, o = umc`

The service named `iptv` is a policy-driven service with the default policy that forwards any traffic. The `iptv` service allows the IPTV subscriber to allocate bandwidth. The `iptv` service should be linked to the action congestion point so that ACP can perform CAC. The bandwidth and congestion point for the service are specified in the Admission Control tab of the SSP Service pane.

- `serviceName = tunnel, o = Services, o = umc`

The tunnel service is a script service that creates an LSP tunnel. The script service must specify an idle timeout value and a substitution for the IPTV_LSPTunnel_ingress attribute (which is the IP address of the LSP tunnel ingress). The script service uses the `getCommandChannel` method in the `ServiceSessionInfo` interface to provide the command channel on which to send commands to the network device (such as a Juniper Networks router) that is attached to the service session. For the script service to work, you must copy the `/iptv/lib/iptv.jar` file to the `/sae/var/run` directory.

For information about the `ServiceSessionInfo` interface, see the script service documentation in the SDX software distribution in the folder `SDK/doc/sae` or in the SAE core API documentation on the Juniper Networks Web site at

<http://www.juniper.net/techpubs/software/management/sdx/api-index.html>

For detailed information about configuring services, see *SDX Services and Policies Guide, Chapter 1, Managing Services*. For detailed information about configuring policies, see *SDX Services and Policies Guide, Chapter 5, Configuring and Managing Policies*.



NOTE: The SAE and JUNOSe routers communicate using the Common Open Policy Service (COPS) protocol. The sample data supports COPS XDR mode.

To use the sample data with COPS-PR mode, you must perform one of the following tasks:

- Define another IPTV subscriber that can start the service named `iptv`.
- Define another service (for example, `iptv2`) so that the same client can start two services.

Configuring ACP as an External Plug-In for the IPTV Application

To configure an external plug-in for the SAE, see *SDX Subscribers and Subscriptions Guide, Chapter 5, Configuring Authorization and Accounting Plug-Ins*.

Configure the object reference with the following value:

- `Plugin.acp.objectref = corbaname:: < host > : < port > /NameService#plugin.acp`
 - `< host >` —Name or IP address of the COS name server
 - `< port >` —TCP port

Use the following values for the plug-in attributes:

- `PA_ROUTER_NAME, PA_INTERFACE_NAME, PA_INTERFACE_ALIAS, PA_PORT_ID, PA_SERVICE_NAME, PA_EVENT_TIME, PA_SESSION_ID, PA_NAS_IP, PA_DOWNSTREAM_BANDWIDTH, PA_UPSTREAM_BANDWIDTH, PA_SERVICE_SESSION_NAME, PA_EVENT_TIME_MILLISECOND, PA_INTERFACE_SPEED, PA_SUBSTITUTION`

Configuring Event Publishers for the IPTV Application

Configure the SAE to publish the following types of events to ACP:

- Global service tracking
- Global interface tracking

Identify the instance of ACP by the name of the host on which you configured it. For example:

- `Service.tracking.plugins = acp`
- `Interface.tracking.plugins = acp`

Configure the SAE to publish the global user tracking events to the NIC; for example: `User.tracking.plugins = nic`.

For information about configuring event publishers, see *SDX Subscribers and Subscriptions Guide, Chapter 7, Configuring Authorization and Accounting Plug-Ins*.

Configuring the NIC as an External Plug-In for the IPTV Application

To configure an external plug-in for the SAE, see *SDX Subscribers and Subscriptions Guide, Chapter 6, How to Configure SAE Plug-Ins*.

Configure the object reference with the following value:

`Plugin.nic.objectref = corbaname:: < host > : < port > /NameService#nicsae/saePort`

- `< host >` —Name or IP address of the COS name server
- `< port >` —TCP port

Use the following values for the plug-in attributes:

- `PA_ROUTER_NAME`, `PA_SESSION_ID`, `PA_USER_TYPE`, `PA_USER_DN`, `PA_USER_IP_ADDR`

Configuring ACP for the IPTV Application

To configure ACP for the sample IPTV application, you must configure ACP as you would normally for backbone mode and enable state synchronization. For more information about configuring ACP, see *SDX Application Library Guide, Chapter 14, Providing Admission Control with ACP*.

In addition to the normal ACP configuration, you must perform these tasks:

1. Defining ACP Properties for the IPTV Application on page 124
2. Defining the Sample Congestion Points for the IPTV Application on page 125

Defining ACP Properties for the IPTV Application

To configure ACP for the sample IPTV application, you must define certain ACP properties. To do so with SDX Admin:

1. Access SDX Admin.
2. In the navigation pane, highlight the entry *I = config, I = ACP, ou = staticConfiguration, ou = Configuration, o = Management, o = umc*.
3. Click the Main tab in the ACP Configuration pane.
4. In the Property field, include these entries so that the NIC proxy in ACP can look up the SAE using the DN.

```
/NicProxy.IPTV/nic.server = /realms/dn/B1
/NicProxy.IPTV/nic.keytype = Dn
/NicProxy.IPTV/nic.valuetype = SaeId
/NicProxy.IPTV/nic.expectmultiple = false
```

For information about NIC proxies, see *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 7, Configuring Applications to Communicate with an SAE*.

5. In the Property field, include these entries to specify how ACP manages the congestion points for the IPTV application and to specify the ACP configuration namespace.

```
Application.iptv.tunnelaction.retryInterval = 5
Application.iptv.tunnelaction.timeout = 10
Application.iptv.tunnelaction.maxRetry = 2
Application.iptv.tunnelaction.waitTime = 60
Application.iptv.tunnelaction.nicNamespace = /<-acpnamespace->/NicProxy.IPTV
```

Application.iptv.tunnelaction.retryInterval

- Time interval at which ACP tries to create, delete, or modify the LSP tunnel.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.retryInterval = 5

Application.iptv.tunnelaction.timeout

- Maximum time ACP waits to create, delete, or modify the LSP tunnel.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.timeout = 10

Application.iptv.tunnelaction.maxRetry

- Maximum number of retry times for creating, deleting, or modifying the LSP tunnel.
- Value—Number
- Default—Application.iptv.tunnelaction.maxRetry = 2

Application.iptv.tunnelaction.waitTime

- Time to wait before re-creating the LSP tunnel after an interface tracking event reports that the LSP tunnel is down.
- Value—Number of seconds in the range 0–2147483647
- Default—Application.iptv.tunnelaction.waitTime = 60

Application.iptv.tunnelaction.nicNamespace

- Name of the object that contains the ACP configuration data for the application.
- Value—/ < ACP namespace > /NicProxy.IPTV
- Guidelines—Replace < -acpnamespace- > with the ACP configuration namespace. This object appears in *l = ACP, ou = staticConfiguration, o = Management, o = umc*.
- Default—Application.iptv.tunnelaction.nicNamespace = / < -acpnamespace- > /NicProxy.IPTV
- Example—Application.iptv.tunnelaction.nicNamespace = /jacp/NicProxy.IPTV

Defining the Sample Congestion Points for the IPTV Application

Define the sample network congestion points and the sample congestion point for the service named iptv by importing the */iptv/conf/ldap/congestionPoint.ldif* file from the installation directory (*/opt/UMC* by default) using an LDAP browser. You can then modify the following congestion points with SDX Admin.

- The congestion points for the iptv service are added under *o = congestionPoint, o = umc*.
- The network congestion points are added under *o = admissionControl, o = umc*.

The *interfaceName = iptv, orderedCimKeys = client@ERX, o = admissionControl, o = umc* object contains a link to the application. For the link to work, you must copy the */iptv/lib/iptv.jar* file to the */acp/var/run* directory.

The sample data for the action congestion point includes the following parameters for managing the LSP tunnel's bandwidth and endpoint information based on the sample application:

- CP_LSP_Egress_IP—Egress IP address for LSP tunnel
- CP_LSP_Initial_Bandwidth—Initial bandwidth for LSP tunnel (in bps)
- CP_LSP_Max_Bandwidth—Maximum bandwidth for LSP tunnel (in bps)
- CP_LSP_Incremental_Bandwidth—Incremental bandwidth for LSP tunnel after the threshold is triggered (in bps)
- CP_LSP_Name—LSP tunnel name

- CP_LSP_DecrementBandwidth_Threshold—LSP decrement bandwidth threshold in percentage
- CP_LSP_IncrementBandwidth_Threshold—LSP increment bandwidth threshold in percentage

The SDX software can trigger changes to the bandwidth of the LSP tunnel based on a given threshold. For example, bandwidth can be increased by 50 Mbps if 90% of the initial bandwidth is utilized by setting these parameter values:

```
CP_LSP_Incremental_Bandwidth = 50000000
CP_LSP_IncrementBandwidth_Threshold = 0.9
```

Your parameters can differ if you write your own application.

For information about configuring congestion points, see *SDX Application Library Guide, Chapter 14, Providing Admission Control with ACP*.

Configuring the NIC for the IPTV Application

To configure the NIC for the sample IPTV application:

- Configure a NIC proxy. See *SDX Network Guide: SAE, Juniper Networks Routers, and NIC, Chapter 7, Configuring Applications to Communicate with an SAE*.
- Add the NIC configuration entry `l = OnePopAllRealms, l = NIC, ou = staticConfiguration, ou = Configuration, o = Management, o = umc` by importing the `/iptv/conf/ldap/nic.ldif` file from the installation directory (`/opt/UMC` by default) using an LDAP browser.

This entry contains a special DN to SAE ID mapping. The NIC host configuration namespace must point to this entry.

Running the Sample IPTV Application

To run the sample IPTV application, you must:

1. Install the omniORB and SMCpython packages from the SDX software distribution. See the *SDX Getting Started Guide* for complete information about installing the SDX core software.
2. Start the SAE, the NIC, and ACP, and establish the router connections.
3. Log in to the IPTV server as a static interface user on the JUNOS routing platform, and log in as the IPTV subscriber on the JUNOS router.
4. In the `/bin` directory, compile the SAE IDL by running `compileIDL`.
5. Include the omniORB library (`/opt/UMC/omni/lib` by default) in the `LD_LIBRARY_PATH` environment. For example:

```
export LD_LIBRARY_PATH=/opt/UMC/omni/lib:$LD_LIBRARY_PATH
export PATH=/opt/UMC/python/bin:$PATH
```

6. In the */bin* directory, run the *startIPTVService1.py* Python script.
7. In the */bin* directory, run the *startIPTVService2.py* Python script.
8. In the */bin* directory, run the *stopIPTVService2.py* Python script.
9. In the */bin* directory, run the *stopIPTVService1.py* Python script.

